Biasing Restricted Boltzmann Machines using Gaussian Filters to Learn Invariant Visual Features

Arjun Yogeswaran and Pierre Payeur
School of Electrical Engineering and Computer Science
University of Ottawa
Ottawa, Canada
ayoge099@uottawa.ca, ppayeur@uottawa.ca

Abstract— **Advances in unsupervised learning have allowed the efficient learning of feature representations directly from large sets of unlabeled data instead of using traditional handcrafted features. However, improving algorithms to increase the quality of these representations in the absence of labeled data is still an area of active research. This paper evaluates visual features learned through unsupervised learning, specifically comparing regularization and preprocessing methods using Gaussian filters on a single-layer network. Using the restricted Boltzmann machine as the unsupervised learning mechanism, features emerging through training on natural videos, with different biasing and preprocessing based on Gaussian filters, are compared by metrics to measure invariance as well as classification performance on standard datasets. When Gaussian filters are convolved with adjacent hidden layer activations from a single example during training, topographies begin to emerge where adjacent features become tuned to slightly varying stimuli. 1D, 2D, and 3D topographies are compared. When a Gaussian low-pass filter is applied to activations from a single hidden node across frames drawn from video, features that are more invariant to transformations are produced. Finally, when Gaussian filters are applied to the visible nodes, images become blurrier; learning from these images also leads to invariant features. The networks are trained using the Hollywood2 video dataset, and tested on image classification of the static CIFAR-10 and STL-10 datasets. To prove that the improvements are independent of the dataset, the networks are shown to produce similar results when trained on the CIFAR-10 dataset. The induction of topography or simple image blurring via Gaussian filters during training produce better discriminative features as evidenced by the consistent and notable increase in classification results that they produce. Also, in the visual domain, invariant features are desirable such that objects can be classified accurately despite transformations. It is found that most of the compared methods produce more invariant features, however, classification accuracy does not correlate to invariance.**

Keywords— unsupervised learning; Gaussian filter; restricted Boltzmann machine; topography; image classification; representation learning; feature representation; object recognition; invariant feature.

I. INTRODUCTION

Recent advances in unsupervised learning mechanisms have allowed increased performance in a variety of domains, including visual object classification. The importance in this performance increase is that these learning mechanisms learn powerful feature representations directly from unlabeled data. An effective way to assess these methods and their properties is to test them on single-layer networks, as opposed the multi-layered networks which are commonly used for image classification, such that the influence of network architecture is minimized and the actual performance of the learning mechanism can more easily be isolated and compared. Coates *et al.* [1] structured their comparison similarly by comparing single-layer networks and their classification performance on standard image classification datasets under a variety of parameters. That study outlined the efficacy of several single-layer techniques at learning discriminative features from raw pixel data. Modeled after that study, our current research aims to compare different regularization and preprocessing techniques also on a single layer in order to boost image classification performance through generating better features. In addition, the properties of these features, which are more difficult to interpret at higher levels in a multi-layer network, may lead to further understanding of their role in object recognition.

An important aspect of learning is the ability of a system to be robust enough to generalize between slightly different versions of the same stimulus, but specific enough to discriminate between the desired stimulus and other stimuli. At the heart of this particular balance lies the concept of invariance. Typically, unsupervised learning techniques will learn visual features which are not particularly invariant to transformations. Increasing a feature's tolerance to such transformations is considered making it more invariant and a more robust datapoint for classification. Architectures such as the convolutional neural network [2] use hard-wired translational invariance to achieve robustness. Though effective, this limits the network to tolerating only one type of transform. Since unsupervised learning methods can learn features from natural data, it is a natural progression to also learn invariances from the same data.

Regularization is the process of biasing a function based on external information with the goal of producing more desirable results. Restricted Boltzmann machines (RBM) [3] have benefitted from regularization to induce sparsity, resulting in the production of more discriminative features, such as physiologically-consistent Gabor-like features when trained on natural image data. Topographic Independent Component Analysis (TICA) [4] uses regularization to create topography in the learned features.

Gaussian filters are at the core of many machine vision techniques, often to reduce noise or high frequency effects, incorporate information from neighboring regions, or soften edges depending on how they are applied. In unsupervised learning, they have been considered for inducing topography, falling under the idea of incorporating information from neighboring regions [5]. Temporal low-pass filtering has been used to learn slow-changing features [6], therefore, Gaussian filters are a viable candidate for implementing the low-pass filter. Generally, incorporating additional information and filtering can improve feature learning, and the Gaussian filter provides exactly those characteristics. As a result, they are a compelling topic to link together different biasing techniques in unsupervised learning.

Le *et al*. created a deep network that used neighborhoods of features found using TICA to achieve invariance en route to state-of-the-art results on a variety of object recognition benchmarks [7]. Zou *et al*. [8] learn slow-changing features for the purpose of invariance, and lower error rates are produced in image classification tasks. These results show that performance can increase over hard-wired invariance when features are learned from the data, thus giving a real reason to explore and quantify methods to achieve this result.

This work characterizes the effectiveness of regularization and preprocessing methodologies, specifically based around the Gaussian filter, at learning discriminative-yet-invariant features by a set of metrics as well as classification performance on standard datasets. The training dataset will be the Hollywood2 video dataset [9] but the testing datasets will be the static CIFAR-10 [10] and STL-10 [1] image classification datasets. Training on one dataset and testing on another in the visual domain has been shown to work effectively [8]. To show the results are not contingent on the dataset, the same networks will also be trained on the CIFAR-10 dataset and compared. To reduce variability when comparing techniques, an RBM with fixed hyperparameters is used as a singular architecture on which different regularizations are tested.

II. Related Work

Learning invariance often occurs by either supplementing temporal information from video or dependence information from images. Using temporal information from video is based on the idea that the transformations that features undergo in video can be captured such that the network can be invariant to them. Földiák was among the first to use a temporal consistency component for learning invariance in a network from sequences of data [6]. The network groups similar features together based on a learning principle coupled with the patterns found in natural motion. In order to track activation over time, Földiák introduces a trace variable, $\bar{y}$, which takes a running average of the activation, y, subject to (1).

$$\bar{y}^{(t)} = (1-\delta)\bar{y}^{(t-1)} + \delta y^{(t)} \tag{1}$$

where δ measures the influence of the current activation on the running average, and y is the current activation of the unit. This low-pass filtering suppresses fast-changing features while preserving slow changing features. The method is only applied on synthetic sequences, yet shows that learning invariance by capturing slow changing features is possible. Einhäuser *et al*. [11] introduce a larger-scale model and apply it to real-world video with online learning using an adapted trace variable. It was found that feature orientations change more slowly than their position, thus, its nodes become tuned to features of similar orientation but varying position. This shows that translation invariance can be learned using temporal structure in natural video. Zou *et al*. [8] use temporal slowness in conjunction with a deep belief network to show that unsupervised training can learn increasing invariances as the network aggregates layers. Building upon the auto-encoder, the cost function is regularized with a temporal difference cost, which encourages the result of pooling nodes in a neighborhood to remain similar between time steps. With a neighborhood size of two, the network learns pairs of features which their respective pooling node becomes invariant to.

When using static images to learn invariance, the idea of overlapping information or dependence is used as supplementary information. TICA [4] creates a smoothly-varying topographic relationship between all of the components, where components outside of a neighborhood are independent, but components within a neighborhood are not. Similarly, Goh *et al*. [5] apply a regularizer to introduce 2D topography into an RBM, learning invariant color features that vary smoothly over the hidden layer. It is also claimed that sparsity and topography create more invariant features. That claim is tested in this work.

III. Background

A. Restricted Boltzmann Machine

The restricted Boltzmann machine(RBM) [12], shown in Fig. 1, is an undirected bipartite network which uses its hidden layer to represent input data from the visible layer. It is an energy-based model, and calculates the energy of the joint configuration of visible nodes and hidden nodes by (2).

$$E(v,h) = -a'v - b'h - h'Wv \tag{2}$$

where v and h are the visible and hidden node states, respectively, a and b are the visible and hidden biases, respectively, and W are the symmetric weights connecting the hidden and visible nodes.

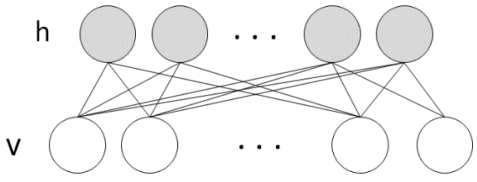

Fig. 1. Restricted Boltzmann machine.

Equation (3) determines the probability that a binary hidden node is on, given the visible vector. To deal with image data, the visible vector is modeled using linear nodes. Equation (4) determines the visible node given the hidden vector.

$$P(h_j = 1 \,|v) = sigmoid(b_j + \textstyle\sum_i \; w_{ij} v_i) \tag{3}$$

$$P(v_i = v\,|h) = \mathcal{N}(v|a_i + \sum_j w_{ij}h_j\,, 1) \quad (4)$$

where h_j is the j^{th} hidden node, v is the visible node vector, b_j is the bias of the j^{th} hidden node, a_i is the bias of the i^{th} visible node, w_{ij} is the weight connecting the i^{th} visible node, v_i, and h_j, $\mathcal{N}(\mu, \sigma^2)$ is a probability density of Gaussian distribution with mean μ and standard deviation σ. Since the image data is normalized, unit variance is used.

Training is accomplished using contrastive divergence (CD), and involves lowering the energy for preferred configurations of hidden and visible nodes, and raising the energy for undesirable configurations [12]. The training alternates between the positive phase and negative phase, where the positive phase samples the hidden state, h^+, and the visible state, v^+, from the data while the negative phase produces the reconstructions of the hidden state, h^-, and the visible state, v^-. The weight update is defined as:

$$\Delta w_{ij} = \gamma[< v_i^+ h_j^+ > - < v_i^- h_j^- >] \quad (5)$$

where γ is the learning rate, and <.> is the average over a number of samples.

Sparsity has been shown to increase discriminative power and optimize RBM representation of data by forcing only a subset of nodes to represent presented data. Lee *et al.* [3] specifies a sparsity target and adds a regularization term to encourage activation with the target frequency by increasing or decreasing the bias.

B. Gaussian Filters

The purposes of the Gaussian filter in this work are conceptually varied, despite obvious similarities. In the case of the topographic RBM, the Gaussian filter incorporates information from local hidden nodes to induce a dependence thus influencing nodes to develop properties similar to its neighbors. In the case of temporal low-pass filtering, the Gaussian filter serves to remove fast-changing signals such that only constant features, despite mild transformations, cause activation. For the purpose of image blurring, the Gaussian filter serves to soften edges such that the network learns blurrier features. The effect of the sigma value, which represents the standard deviation of the Gaussian function, determines the magnitude of the induced effect. Note that these filters are normalized such that their gain stays at 1.

IV. Compared Biasing Methods

The idea of this work is to compare the application of Gaussian filters to induce different effects during unsupervised learning of visual features in an RBM. The compared biasing methods will be: regular RBM, topography, temporal low-pass filtering, and input blurring.

When training in a batch, the weighted sums in (3) and (4) are performed by matrix multiplication. The visible input matrix contains all of the training data to be used in the current batch, where each row represents a training pattern, or image patch in this work, and each column represents a visible node. The weight matrix contains the weights connecting each visible node to each hidden node. The hidden activation matrix is calculated by multiplying the visible matrix and the weight matrix, resulting in a matrix of activations with each row representing a pattern and each column representing a hidden node. Fig. 2 shows how a Gaussian filter is applied in the visible matrix and hidden activation matrix for each method which will be compared.

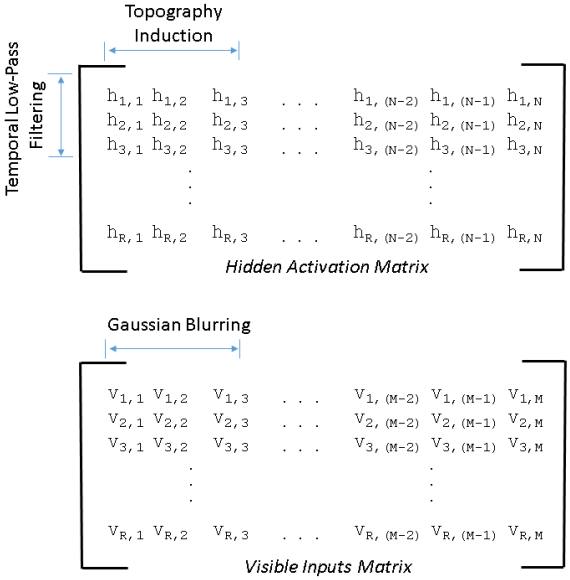

Fig. 2. Application of convolution, where h are hidden node activations, v are visible inputs, M is the # of visible nodes, N is the # of hidden nodes, and R is # of patterns. Gaussian filters for topography induction and for temporal low-pass filtering are applied to the hidden activation matrix in the shown directions (top), while the input blurring filter is applied to the visible inputs matrix (bottom).

A. Regular RBM

An RBM with regularization only to induce sparsity [3] is used as a control and will be compared as a baseline. This will be referred to as the regular RBM.

B. Topography

Topography is induced in the RBM by regularization. By ordering the nodes for a 1D topography, or arranging the hidden nodes in a grid for a 2D or 3D topography, neighboring nodes can be determined. Applying a Gaussian filter to hidden node activations at each example, each node incorporates information about its neighboring nodes. Applied during learning, adjacent nodes develop slightly different features that gradually vary across the grid.

Assuming batch training, a Gaussian filter is applied among adjacent hidden nodes exposed to the same pattern in the activation matrix. The positive phase activations of the hidden nodes are modified by (6) as found in [5].

$$\hat{h}_j^{(k)} = \sum_n^N h_n^{(k)+} \omega(j,n) \quad (6)$$

where $\hat{h}_j^{(k)}$ is the topography-induced positive activation of hidden node j at pattern k, $h_n^{(k)+}$ is the positive phase hidden activation of hidden node n at pattern k, and the neighborhood function, ω, is a set of fixed neighborhood weights which

controls the impact of the surroundings on each activation. ω is set to a Gaussian function. A 1x3, 3x3, and 3x3x3 kernel is used for the 1D, 2D, and 3D topographies, respectively.

Fig. 3 depicts this by showing the results of training a RBM when inducing 2D topography, where each element in the 20x20 grid is a visualization of the feature that each hidden node learns. For example, a vertical feature node responds best to vertical edges, a colored feature node responds best to the displayed color, and a patterned feature node responds best to the displayed pattern.

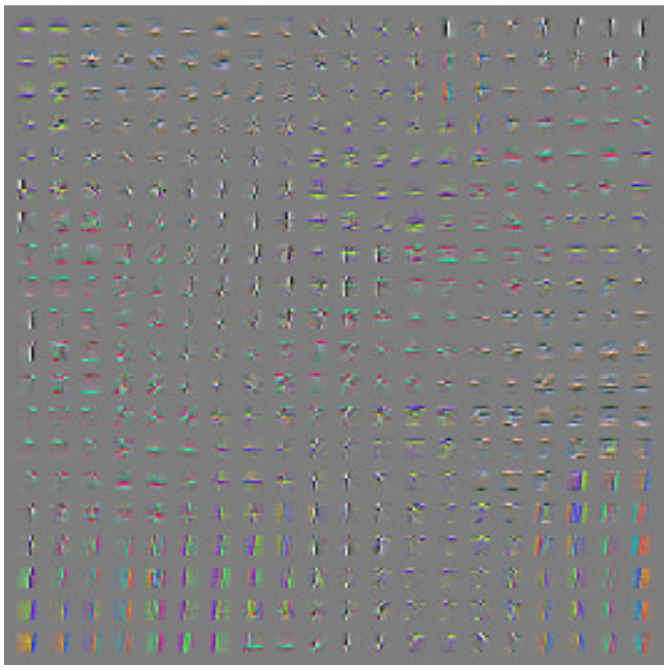

Fig. 3. 20x20 grid of features produced by 2D topography in an RBM.

C. *Temporal low-pass filtering*

Invariant representations may also be learned from video. Prior work has indicated that temporal low-pass filtering on activations during learning of transformation sequences gives rise to invariant features [6, 13].

Since consecutive patterns, or rows in the visible inputs matrix, are the same image patch in consecutive time frames extracted from Hollywood2, a temporal continuity effect is induced during batch training by applying a Gaussian filter to the hidden activation matrix perpendicular to the way it was in the topography organization, meaning blurring among the same hidden node exposed to adjacent patterns as shown in (7). This transposes the ideas presented in the discussed papers to the RBM framework used in this work.

$$\hat{h}_j^{(k)} = \sum_{r=1}^{R} h_j^{(r)+} \omega(k, r) \tag{7}$$

where this ω represents the Gaussian weighting of the activation that different patterns produce on the same hidden node, which is a type of temporal low-pass filter.

D. *Input blurring*

Though blurring could fall under preprocessing, it is addressed as a method used for comparison. Image patches are blurred using a Gaussian filter with kernel width 3 and varying sigmas, using (8), before being passed to the RBM.

$$v_i^{(k)} = \sum_{m=1}^{M} v_m^{(k)} \omega(i, m) \tag{8}$$

where $v_i^{(k)}$ is the visible node i at pattern k being blurred, $v_m^{(k)}$ is a neighboring visible node m at pattern k, and the neighborhood function, ω, is the Gaussian filter

V. Experiment Design

A. *Preprocessing*

In both training and testing, the color patches are contrast normalized and whitened, as these are common techniques to reduce redundant information [1].

B. *Training*

Primarily, the Hollywood2 action dataset [9] was used for training. It is a natural video dataset containing color video clips taken from Hollywood films; two examples are shown in Fig. 4.

Fig. 4. Examples from Hollywood2 dataset.

The dataset contains enough real-world scenery, image variation, and motion to suit the training algorithms that use static images (regular, topographic, input blurred methods) as well as the ones that make use of video (temporal low-pass method). To efficiently use the video dataset, motion is found via background subtraction between adjacent frames, selecting the position with the highest difference, and using the image patch at that location as a training example. The sequence lasts for 5 frames until a new position is found through the same process. Sequential time frames are consecutive training examples such that the temporal low-pass filter can operate as described. The network was trained for 300 epochs, on 10,000 5-frame sequences, divided into batches of 100. As an additional experiment, the methods were also trained on CIFAR-10 [10] to see if the same effects persisted across training datasets. 50,000 random patches were used as training data, and the other training parameters remained the same.

C. *Testing*

As a measure of how effective the techniques are, the trained networks are applied on two datasets, CIFAR-10 [10] and STL-10 [1], which are composed of real-world images containing objects belonging to various classes. Classification

accuracy is reported according to the standard testing procedure for each dataset. Samples of CIFAR-10 are shown in Fig. 5.

Fig. 5. Some examples of images found in CIFAR-10, with labels of dog, automobile, and airplane (left to right).

The classification procedure follows the one outlined in [1]. The training example is transformed into a set of subpatches, each of which are passed through the RBM to generate a set of feature vectors representing the entire image. Experiments in this work use a stride of 1, which means no patches are skipped over. Therefore, an image of n-by-n pixels, and an input-patch size of w-by-w, produces an (n-w+1)-by-(n-w+1) representation with K features. y_{ij} denotes the K-dimensional representation extracted at position (i,j). A simple pooling mechanism is implemented by dividing the image into 4 quadrants, and the feature vector is reduced from a (n-w+1)-by-(n-w+1)-by-K representation to a 2-by-2-by-K representation by summing the y_{ij}'s in each quadrant. A standard linear classifier is used for classification with the summed feature vectors and the associated label.

D. Invariance Metric

The metric proposed by Goodfellow *et al.* [14] simplifies each network's invariance properties into a comparable value. The idea is to compare a feature's activation when presented with transformed optimal stimuli versus random stimuli. The local trajectory T(x) is a set of stimuli that are semantically similar to a reference stimulus x:

$$T(x) = \{\tau(x,\gamma) | \gamma \in \Gamma\} \tag{9}$$

where τ(x,γ) transforms a stimulus x into a new stimulus, and γ is a transformation which comes from a set of possible transformations, Γ. For example, these transformations could be all translations within 5 pixel radius, or rotations within 45 degrees. The robustness of a hidden node is calculated by the firing of a hidden node when applied to local trajectories around inputs which maximally activate that node:

$$L(j) = \frac{1}{|Z|}\sum_{z \in Z}\frac{1}{|T(z)|}\sum_{x \in T(z)} f_j(x) \tag{10}$$

where fj(x) is the activation, calculated by (3), of hidden node j when presented with stimulus x. Z is the set of inputs that produce a high activation from that same hidden node, and T(z) is the set of local trajectories around input z. The global firing rate, G(j), is the firing rate of a hidden node when applied to stimuli drawn randomly from a distribution of possible inputs defined for the test. Finally, the invariance score, S(j) is calculated:

$$S(j) = \frac{L(j)}{G(j)} \tag{11}$$

This calculation represents the robustness of the hidden node divided by the global firing rate, where L(j) indicates the node's robustness, while G(j) ensures that the hidden node is selective and not always active. A higher invariance score indicates that the hidden node is more invariant. Since the network may not dedicate all of its resources to encoding invariance information, only the p top-scoring nodes are used in calculating the average score for the network.

VI. Results and Analysis

Experiments were carried out with RBMs using the methodologies detailed earlier: regular, 1D, 2D, and 3D topographies, temporal low-pass and input blurred. All networks are regularized with a sparsity of 0.01 and weight decay of 0.002, with no momentum term and no decaying of any hyperparameters. The Hollywood2 dataset is used for training unless otherwise stated. An 8x8 color receptive field size was used in all tests. An 8x8 field is sufficiently small to yield good classification results, while being large enough that optimal stimuli can be appropriately translated and rotated to generate enough data for the invariance score metric. Any other parameters specific to each method are outlined in the results.

In the 3D topography, to keep the same number of nodes in each dimension, a different number of total nodes than the 1D and 2D topography was chosen for comparison. The largest cube number that is smaller than the comparison number is chosen as the number of hidden nodes for the 3D topography. This amounts to 216, 343, 512, 729, 1000, and 1331 nodes for the 3D topography corresponding to 225, 400, 625, 900, 1225, and 1600 nodes for the 1D and 2D topographies, respectively. Examples of features learned by the regular, temporal low-pass, and input blurred RBMs, with 400 hidden nodes, are shown in Fig. 6, 7, and 8, respectively. The 2D topography is shown in Fig. 3.

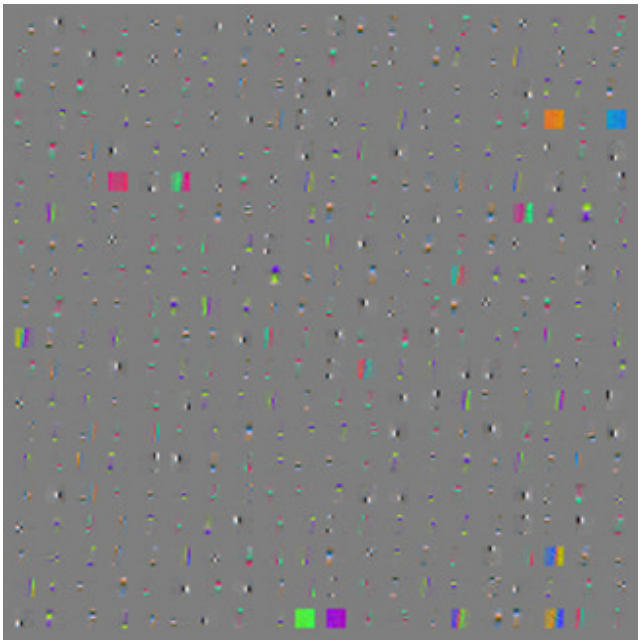

Fig. 6. 20x20 grid of features produced by the regular RBM.

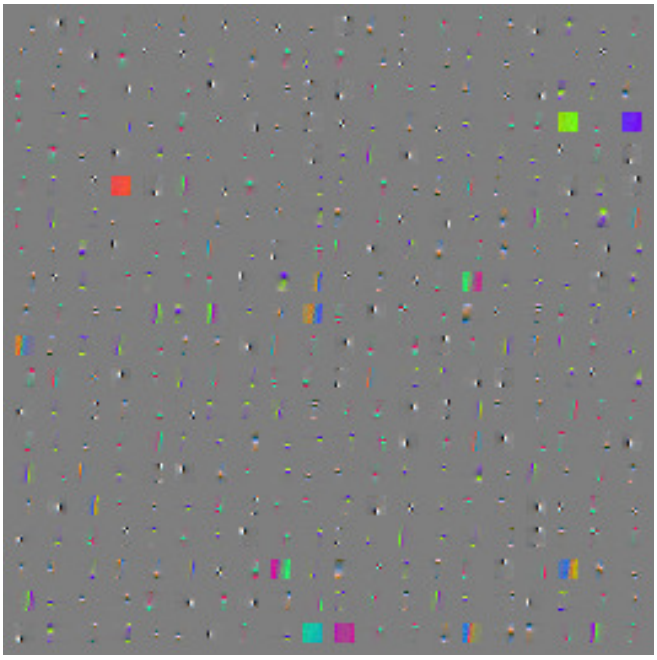

Fig. 7. 20x20 grid of features produced by the temporal low-pass RBM.

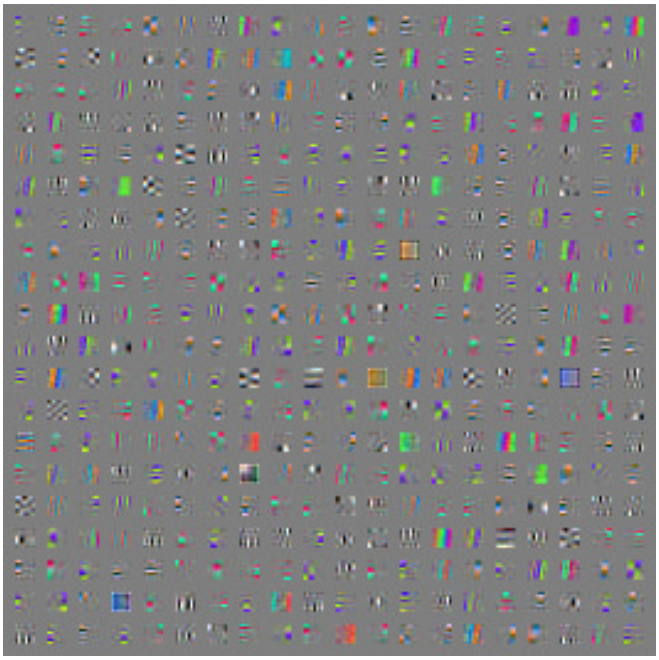

Fig. 8. 20x20 grid of features produced by the input blurred RBM.

The 2D topography contains smooth variations among neighboring nodes in both axes. The temporal low-pass filtered RBM produces very similar features to the regular RBM. The features learned by the RBM with input blurring shows more high frequency features with less localization. The topographical RBM also has more high frequency features with less localization, but they look less clean and isolated; while less pleasant to look at, it may indicate that it has learned a more diverse and complex set of correlations.

A. *Classification*

The methods and their classification accuracies on CIFAR-10 relative to the number of hidden nodes in the network are shown in Fig. 9. Within each method, the sigma with the best CIFAR-10 classification accuracy at 1600 nodes is the representative, determined by evaluations of sigma between 0.0 and 2.0. Here, the input blurring and the topographical methods show a constant improvement over the regular RBM and temporal low-pass filtered RBM at each node count.

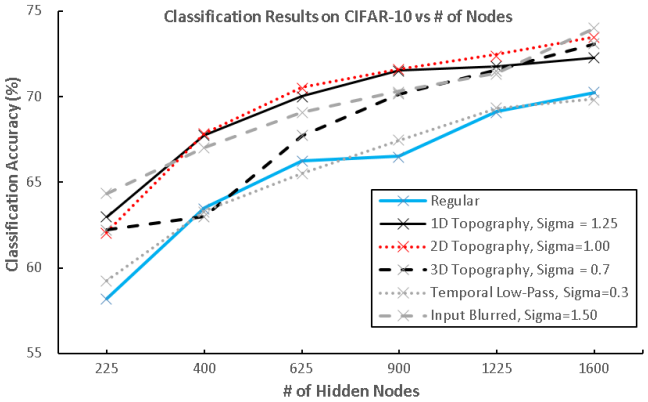

Fig. 9. Best parameters for each method comparing the classification accuracy on the CIFAR-10 test set vs the number of hidden nodes.

To show that these improvements are not due to the Hollywood2 dataset nor the method to select regions of interest, the networks are trained on the CIFAR-10 dataset. The best classification results for each method on CIFAR-10 and STL-10, after being trained on Hollywood2 and CIFAR-10, are shown in Table 1. The temporal low-pass method is not trained on CIFAR-10 since training samples are not temporally related.

TABLE I. BEST CLASSIFICATION RESULTS, WITH 1600 NODES, ON CIFAR-10 AND STL-10, WHEN TRAINED ON HOLLYWOOD2. NOTE THAT THE TOPOGRAPHIC 3D RBM USES 1331 NODES.

Method (RBM)	**Classification Accuracy (%)**			
	Trained on Hollywood2		***Trained on CIFAR-10***	
	CIFAR	***STL***	***CIFAR***	***STL***
Regular RBM	70.26	48.63	73.82	51.49
Topographic 1D (σ=1.25)	72.29	53.05	75.41	52.21
Topographic 2D (σ=1.00)	73.49	**53.63**	**76.24**	54.06
Topographic 3D (σ=0.75)	73.09	51.68	75.65	**54.76**
Temporal low-pass (σ=0.30)	69.85	49.32	N/A	N/A
Input blurred (σ=1.50)	**74.00**	51.50	74.76	52.10

It is visible that the methods achieve similar performance increases relative to the regular RBM regardless of the training dataset. All of the methods perform measurably better than the regular RBM, with the exception of the temporal low-pass filtered RBM. STL-10 contains much less labeled data than CIFAR-10, causing it to suffer from a lower classification rate. However, the values reported are consistent with the performance of single-layer networks using this classification protocol on these datasets [1].

Using basic Gaussian blurring as input preprocessing produces a surprising increase in classification accuracy. So much that it exceeds the regular RBM's performance, when trained on Hollywood2, by 6 percent when compared at 225 nodes and 3 percent at 1600 nodes.

The topographic methods perform best, and classification accuracy is boosted even without pooling, as the features themselves benefit from the shared information within the neighborhood during training. With topography, the grid arrangement is not important after training. Though neighborhood pooling would improve results, it would bleed over into becoming a two-layer network; as a result, it is not discussed here.

Overall, the 1D topography produces similar results to the 2D and 3D topographies. However, it is computationally simpler, since its 1x3 kernel only requires 2 neighbors. Thus, sharing a small amount of information between nodes produces a large improvement in features, and this improvement does not correlate with the number of neighbors, given that the results of the 2D topography typically exceed those of the 3D topography. For these reasons, the 1D topography is a better solution than the other topographic methods when training time is important. Otherwise, the 2D topography is the solution which produces the best results. The 3D topography is the least efficient, since it does not produce much better results than the 2D topography yet requires more computation.

The results show that the convolution of a Gaussian filter with the RBM's activation matrix to incorporate local information from the same example, whether it is applied to input nodes or hidden nodes, provides a notable increase in classification performance. Incorporating temporal information in this manner does not seem to provide any discernible benefit.

B. Invariance

The invariance score, calculated by (11), is used to evaluate how invariant the learned features are. For this score, the methods were trained on grayscale versions of the Hollywood2 dataset. Then, the optimal stimulus for each hidden node was found by searching through all possible line segments, with all possible thicknesses, that fit in the image patch and selecting the one with the highest response. Since most features nodes respond to shapes similar to lines, they were used as the shape by which the optimal stimulus is defined; an example optimal stimulus corresponding to a hidden node feature is shown in Fig. 10.

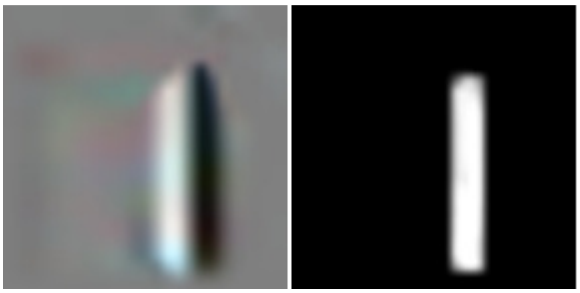

Fig. 10. Hidden node feature (left). Optimal stimulus (right).

Translations of -4 to +4 pixels in both axes, and rotations of -45 to +45 degrees were used as local trajectories according to (9), and the final scores, calculated by (11), were summed together. As in [14], a firing threshold for each node is selected such that its firing rate when applied to random stimuli, G(j), is 0.01. The top 10% of the scores were used to calculate the means, which are reported in Table 2.

TABLE II. INVARIANCE SCORES OF ALL METHODS, WITH 225 NODES (216 FOR TOPOGRAPHIC 3D RBM) TRAINED ON GRAYSCALE VERSIONS OF HOLLYWOOD2.

Method (RBM)	Invariance Score
Regular	22.7
Topographic 1D (σ=1.25)	13.1
Topographic 2D (σ=1.00)	34.4
Topographic 3D (σ=0.75)	48.6
Temporal low-pass (σ=0.30)	25.7
Input Blurring (σ=1.50)	56.4

As shown in the classification results, the topographic and input blurred RBMs outperform the regular RBM and temporal low-pass RBM. However, the invariance scores do not necessarily reflect that invariance is the main reason for this improvement. The topographic 2D and 3D RBMs, as well as the input blurred RBM, have higher invariance scores and higher classification results, however the topographic 1D RBM has a lower invariance score than even the regular RBM, yet still produces higher classification results. The discrepancy in invariance scores between the 1D and 2D RBM is significant, but that does not translate into a significant discrepancy in classification accuracy. Therefore, the ability to form a good representation goes far beyond the ability to generate invariant features.

There are many reasons as to why these techniques produce better representations. With input blurring, edges are softened, resulting in the network learning features that cover a larger area. Thus, it has invariance to small transformations of a sharp feature within that area. With the induction of topography by locally sharing information, neighboring nodes learn similar features. This forces features into interpolating between neighbors, resulting in a higher response to a larger range of transformations. Overall, input blurring and topography are simple methods to improve the learning of visual feature representations.

VII. CONCLUSION

This paper performed a comparison between the biasing effects of Gaussian filters in unsupervised training of an RBM to evaluate their relative performance when it comes to learning invariant detectors for image features. In addition to improvements in classification results, an invariance metric was shown to see how invariant the features are to transformations of their optimal stimulus.

The simple blurring of training images produces a surprising increase in classification accuracy and invariance metrics, sometimes outperforming the other techniques. The biasing to induce topography also quite obviously produces better classification results than without biasing. Temporal low-pass filtering with video does not produce features that are any better

at classification than the RBM without biasing, yet their invariance measure is higher.

The comparison between invariant properties and classification results of different learning methodologies for unsupervised networks produces tangible evidence that, despite differences in the approach, improved features can be achieved by biasing the network appropriately. Invariance is an important property to allow compact representation, and strike a balance between generalization and specificity. However, improved classification results, and the idea of learning good features, should not necessarily be credited to invariant features since there are more factors involved, evidenced by the increased classification performance of the 1D topography despite a lower invariance score.

Overall, this work shows that simple preprocessing and regularization methods are capable of balancing the properties of invariance and achieving good features.

ACKNOWLEDGMENT

Authors acknowledge the support from Natural Sciences and Engineering Research Council of Canada and Ontario Ministry of Training, Colleges and Universities toward this research.

REFERENCES

[1] A. Coates, H. Lee, and A. Ng, "An analysis of single-layer networks in unsupervised feature learning", Proc. International Conference on Artificial Intelligence and Statistics (AISTATS), 2011, pp. 215-223.

[2] A. Krizhevsky, I. Sutskever, and G. Hinton, "Imagenet classification with deep convolutional neural networks", Proc. Advances in Neural Information Processing Systems (NIPS), 2012, pp. 1097-1105.

[3] H. Lee, C. Ekanadham, and A. Ng, "Sparse deep belief net model for visual area V2", Proc. Advances in Neural Information Processing Systems (NIPS), 2008, pp. 873-880.

[4] A. Hyvärinen, P. Hoyer, and M. Inki, "Topographic independent component analysis", Neural Computation, vol. 13, 2001, pp. 1527-1558.

[5] H. Goh, L. Kusmierz, J.-H. Lim, N. Thome, and M. Cord, "Learning invariant color features with sparse topographic restricted Boltzmann machines", Proc. 18th IEEE Conference on Image Processing (ICIP), 2011, pp. 1241-1244.

[6] P. Földiák, "Learning invariance from transformation sequences", Neural Computation, vol. 3, 1991, pp. 194-200.

[7] Q. Le, M. Ranzato, R. Monga, M. Devin, K. Chen, G. Corrado, J. Dean, and A. Ng, "Building high-level features with large scale unsupervised learning", Proc. International Conference on Machine Learning (ICML), 2012, pp. 81-88.

[8] W. Zou, S. Zhu, A. Ng, and K. Yu, "Deep learning of invariant features via simulated fixations in video", Proc. Advances in Neural Information Processing Systems (NIPS), 2012, pp. 3203-3211.

[9] M. Marszalek, I. Laptev, and C. Schmid, "Actions in context", Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2009, pp. 2929-2936.

[10] A. Krizhevsky, "Learning multiple layers of features from tiny images", Technical Report, University of Toronto, 2009.

[11] W. Einhäuser, C. Kayser, P. König, and K. P. Körding, "Learning the invariance properties of complex cells from their responses to natural stimuli", European Journal of Neuroscience, vol. 15, 2002, pp. 475-486.

[12] G. Hinton and R. Salakhutdinov, "Reducing the dimensionality of data with neural networks", Science, vol. 313, 2006, pp. 504-507.

[13] L. Isik, J. Z. Leibo, and T. Poggio, "Learning and disrupting invariance in visual recognition with a temporal association rule", Frontiers in Computational Neuroscience, vol. 6, 2012.

[14] I. Goodfellow, Q. Le, A. Saxe, H. Lee, and A. Ng, "Measuring invariances in deep networks", Proc. Advances in Neural Information Processing Systems (NIPS), 2009, pp. 646-654.